IN THE CLAIMS

18. (Currently Amended) An arbitration circuit for a computer system, the arbitration circuit adapted to couple with a plurality of slave devices having transactions queued for execution, the arbitration circuit further adapted to signal any the plurality of slave devices to reorder their transactions without signaling a microprocessor of the computer system.

19. (Currently Amended) A computer-implemented method for reordering transactions, the method comprising:

receiving and queuing within a slave device a plurality of transactions for execution; and

signaling the slave device to reorder its transactions without signaling a microprocessor of the computer system that the transactions are being reordered.

20. (New) A method of avoiding deadlock in a computer system having a split-transaction bus and a single-envelope bus bridged by a bus bridge, the split-transaction bus and the single-envelope bus each having at least one master device and one slave device connected thereto, comprising:

storing data from one or more accepted bus transactions;

determining, prior to a request for a bus transaction from a bus transaction requestor, if execution of such bus transaction would cause deadlock based on the stored data; and

responsive to the determination that execution of the bus transaction would cause deadlock, sending a retry signal to the bus transaction requestor.

21. (New) The method of claim 20, wherein storing data comprises storing data identifying a slave device and its current transaction.

22. (New) A method of avoiding deadlock in a computer system having a first single envelope device and a second multiple envelope device coupled to a bus, the method comprising:

storing a state of an accepted bus transaction of a first device; and

responsive to the state of the accepted bus transaction, sending a retry signal to a second device where the state of the accepted bus transaction indicates that execution of the requested bus transaction from the second device would cause deadlock between the first and second devices.

23. (New) The method of claim 22, wherein the state comprises a transaction type, and transaction address, and deadlock is indicated by the accepted bus transaction and the requested bus transaction having conflicting transaction addresses.

24. (New) A method of avoiding deadlock in a computer system having a split-transaction bus and a single-envelope bus bridged by a bus bridge, the split-transaction bus and the single-envelope bus each having at least one master device and one slave device connected thereto, comprising:

receive a request for a bus transaction from the first single envelope device; and

determining, prior to accepting the request, whether execution of such bus transaction would cause deadlock between the first device and the second device;

responsive to the determination that execution of the bus transaction would cause deadlock, sending a retry signal to the first device.

25. (New) An apparatus for avoiding deadlock in a computer system having a split-transaction bus and a single-envelope bus bridged by a bus bridge, the split-transaction bus and the single-envelope bus each having at least one master device and one slave device connected thereto, comprising:

a memory for storing data from one or more accepted bus transactions; and

deadlock avoidance logic coupled to the memory for determining, prior to a request for a bus transaction from a requester, if execution of the bus transaction would cause deadlock based on the stored data, the deadlock avoidance logic adapted to send a retry signal to the bus transaction requester if execution of the bus transaction would cause deadlock.

26. (New) A computing system, comprising:

a first single envelop device;

a second, multiple envelope device;

a bus coupling the first and second devices;

a memory coupled to the bus, and to the first and second devices, for storing data from one or more bus transactions from the first and second devices as requesting devices; and

deadlock avoidance logic coupled to the memory for establishing, prior to a request for a bus transaction from a requesting device, whether execution of a subsequent bus transaction would cause deadlock based on the stored data, the deadlock avoidance logic further adapted to send a retry signal to the requesting device in response to a receiving the subsequent request for execution of the bus transaction that would cause deadlock.